Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A hardware-based multithreaded processor comprising:

a plurality of microengines, each of the microengines comprising:

a control store;

controller logic;

context event switching logic; and

an execution box data path including an arithmetic logic unit (ALU) and a general purpose register set, the execution box performing functions in response to instructions, one of the instructions causing the execution box to selectively load any specified combination of bytes of data within a transfer register associated with one of the plurality of microengines, with a shifted value of an operand that preserves the bytes of data that are not loaded, with loading using an associated bit mask specified in the instruction with each bit of the associated bit mask identifying a different byte of the transfer register.

2. (Cancelled)

3. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a left shift n bits, where n is a number from one to thirty-one.

4. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a left shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

5. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a right shift n bits, where n is a number from one to thirty-one.

6. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a right shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

7. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a left rotate n bits, where n is a number from one to thirty-one.

8. (Previously presented) The processor of claim 1 wherein the instruction further comprises a field that indicates a right rotate n bits, where n is a number from one to thirty-one.

9. (Previously presented) The processor of claim 1 wherein the instruction further comprises an optional token that is set by a programmer and specifies to set arithmetic logic unit (ALU) condition codes based on a result formed in the ALU.

10. (Currently amended) A method of operating a processor comprising:

selectively loading any combination of bytes of data within a register associated with one of a plurality of microengines with a shifted value of an operand, using an associated bit mask specified in an instruction, with each bit of the associated bit mask identifying a different byte of the register; and

clearing the bytes of data that are not loaded.

11. (Cancelled)

12. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a left shift n bits, where n is a number from one to thirty-one.

13. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a left shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

14. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a right shift n bits, where n is a number from one to thirty-one.

15. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a right shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

16. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a left rotate n bits, where n is a number from one to thirty-one.

17. (Previously presented) The method of claim 10 further comprising:

providing a field that indicates a right rotate n bits, where n is a number from one to thirty-one.

18. (Previously presented) The method of claim 10 further comprising an optional token that is set by a programmer and specifies to load arithmetic logic unit (ALU) condition codes based on a result formed in the ALU.

19. (Currently amended) A method of operating a processor comprises:

selectively loading any combination of bytes of data within a register associated with one of a plurality of microengines with a shifted value of an operand, using an associated bit mask

specified in an instruction, with each bit of the associated bit mask identifying a different byte of the register; and

preserving the bytes of data that are not loaded.

20. (Cancelled)

21. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a left shift n bits, where n is a number from one to thirty-one.

22. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a left shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

23. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a right shift n bits, where n is a number from one to thirty-one.

24. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a right shift by an amount specified in a lower five bits of the operand of a previous instruction, where the lower five bits is a number from one to thirty-one.

25. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a left rotate n bits, where n is a number from one to thirty-one.

26. (Previously presented) The method of claim 19 further comprising:

providing a field that indicates a right rotate n bits, where n is a number from one to thirty-one.

27. (Previously presented) The method of claim 19 further comprising an optional token that is set by a programmer and specifies to load arithmetic logic unit (ALU) condition codes based on a result formed in the ALU.